
Data

Intensive

Scalable

Computing for
Science

Julio López

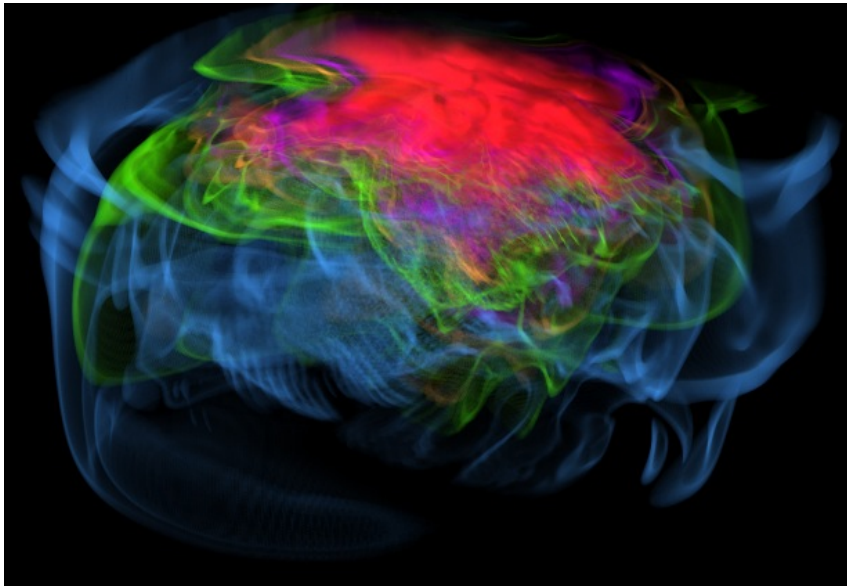
Randal Bryant, Garth Gibson, Greg Ganger

Carnegie Mellon University

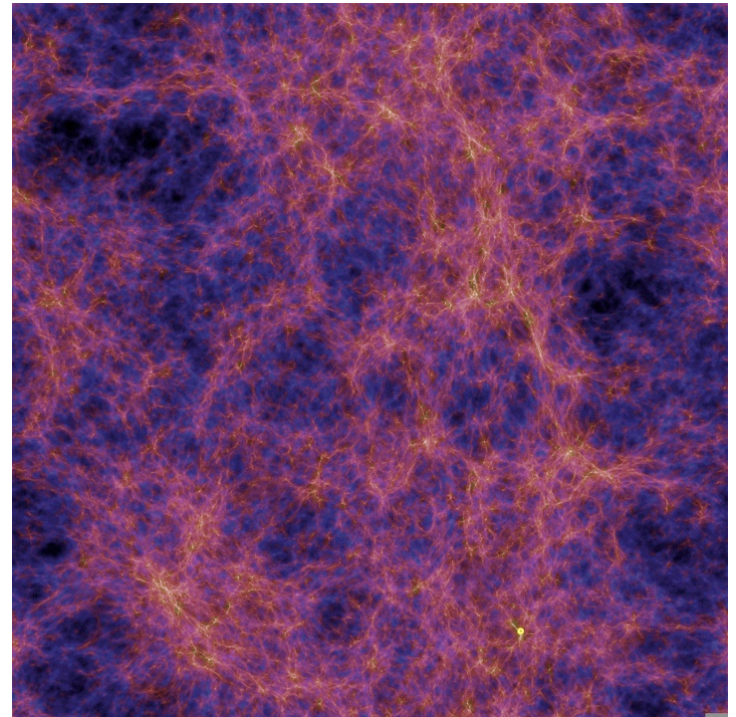
<http://www.pdl.cmu.edu/>

Big Data Sources: Simulations

- Large multi-terabyte simulation datasets: 1-30 TB
- Cosmology simulations soon 10-100 X bigger



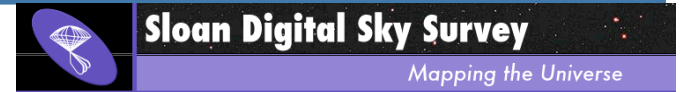
Earthquake simulations



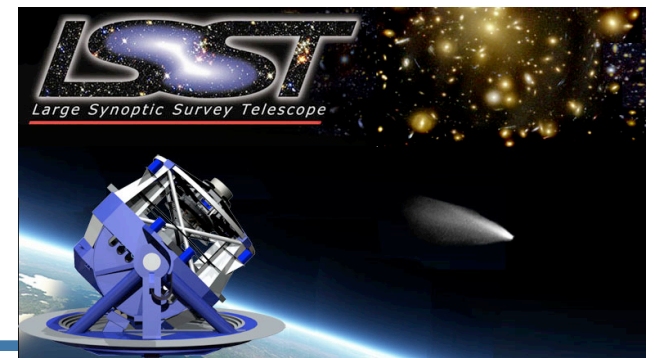
Cosmology

Big Data Sources: Sensors

- Sloan Digital Sky Survey
 - New Mexico telescope: 200 GB/night
 - Latest dataset release: 10 TB
 - 287 million celestial objects
 - SkyServer provides SQL access



- Pan-STARRS
- LSST: Large Synoptic Survey Telescope (2016)
 - 6.4 GB images, 15 TB / night



Big Data Sources: Commerce

- Wal-Mart
- 267 million items/day, sold at 6,000 stores
- HP building them 4PB data warehouse
- Mine data to:
 - Manage supply chain
 - Understand market trends
 - Formulate pricing strategies

Our Data-Driven World

- **Science**
 - Astronomy, seismology, genomics, natural languages ...
- **Humanities**
 - Scanned books, historic documents, ...
- **Commerce**
 - Corporate sales, stock market transactions, airline traffic, ...
- **Entertainment**
 - Internet images, Hollywood movies, music files, ...
- **Medicine**
 - MRI & CT scans, patient records, ...

Why So Much Data?

- We can generate it and get it
 - Automation + Internet
- We can keep it
 - 1 TB Disk @ \$100 (10¢ / GB)
- We can use it
 - Scientific breakthroughs
 - Business process efficiencies
 - Realistic special effects
 - Better health care
- Could we do more?
 - Apply more computing power to this data



Data analytics adds value to the data

Oceans of Data, Skinny Pipes

- Analytics at scale are I/O intensive
- Terabytes: easy to store, hard to move



Time to scan 1 TB		
Disks	MB / s	Time
Consumer	40	7.3 hours
Enterprise	125	2.2 hours
Networks	MB / s	Time
Home Internet	< 0.625	> 18.5 days
Gigabit Ethernet	< 125	> 2.2 hours
Teragrid Connection	< 3,750	> 4.4 minutes

Data-Intensive System Challenge

- For computation that accesses 100 TB in 10 mins
 - Single disk: 2 - 8 GB/min
 - Data distributed over 1000+ disks
 - Assuming uniform data partitioning
 - Compute using 1000+ processors
 - Connected by at least gigabit Ethernet
- System Requirements
 - Lots of disks
 - Lots of processors
 - Located in close proximity
 - Within reach of fast, local-area network

Google's Computing Infrastructure

Barroso, Dean, Hölzle, "Web Search for a Planet: The Google Cluster Architecture" IEEE Micro 2003

- System
 - Millions processors in clusters of ~2000 processors each
 - Commodity parts
 - x86 processors, IDE disks, Ethernet communications
 - Gain reliability through redundancy & software management
 - Partitioned workload
 - Data: Web pages, indices distributed across processors
 - Function: crawling, index generation, search, document retrieval, Ad placement
- A Data-Intensive Scalable Computer (DISC)
 - Large-scale computer centered around data
 - Collecting, maintaining, indexing, computing
- Similar systems at Microsoft, Yahoo, Facebook, Amazon

Challenges of Scale

- Managing thousands of hosts
- Component failures become the steady state
- Distributed resilient apps are hard to write


MapReduce Programming Model

Dean & Ghemawat: “MapReduce: Simplified Data Processing on Large Clusters”, OSDI 2004

- Programming abstraction and runtime support
- Scaling up applications
- Make it easy to use thousands of nodes
- Common application pattern
 - Input: Large unordered collection of unstructured records
 - Process each record
 - Group intermediate results
 - Process groups
- Scalable distributed “GROUP BY” primitive
- Hadoop open source implementation



MapReduce



Input

MapReduce

Input Split

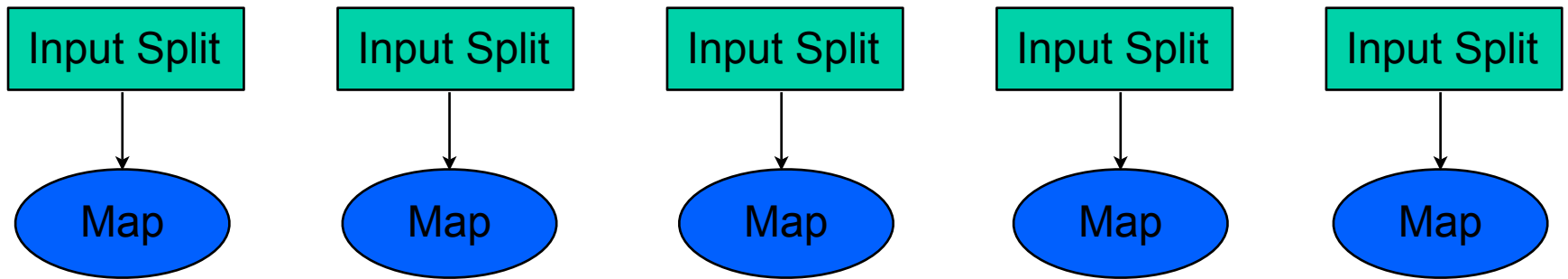
Input Split

Input Split

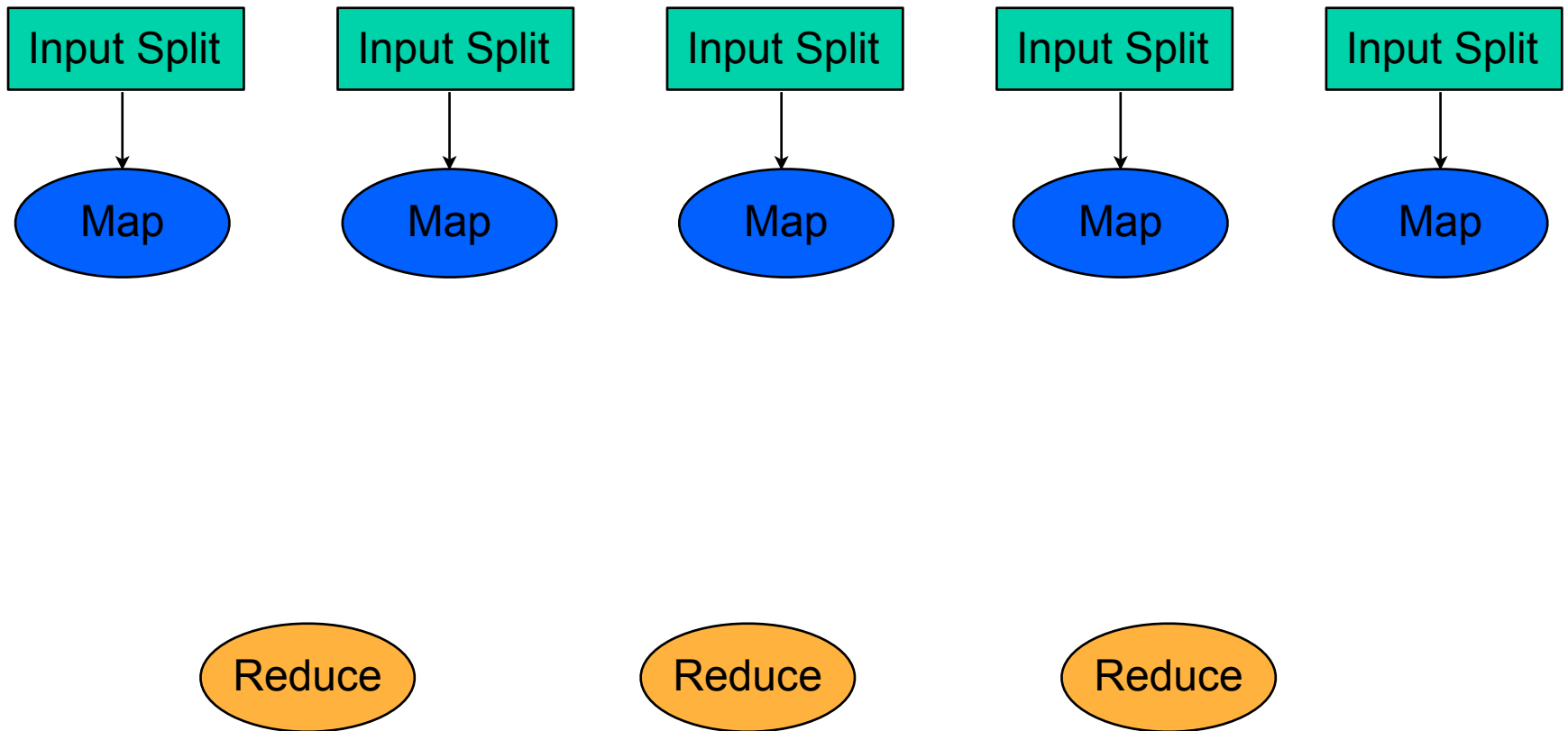
Input Split

Input Split

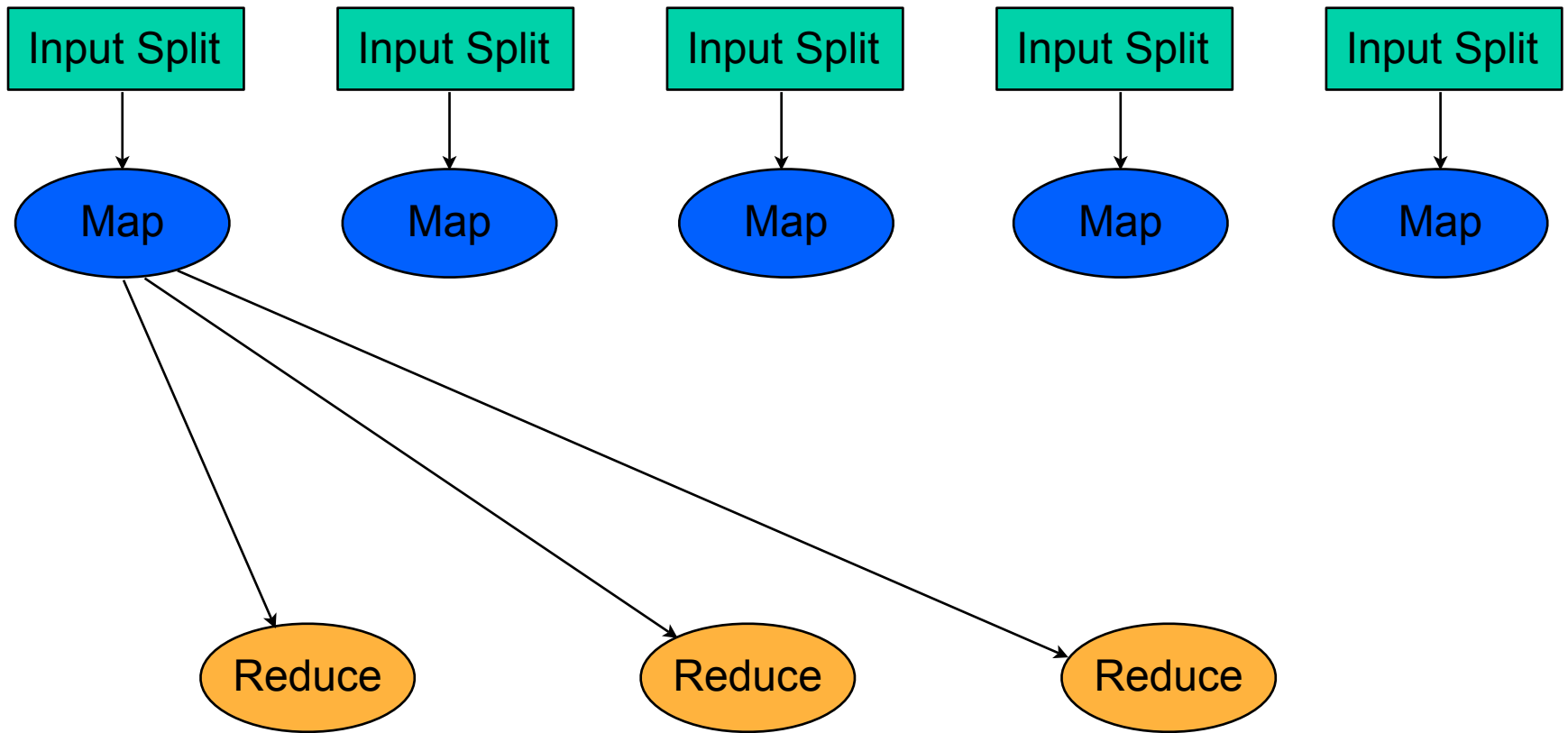
MapReduce



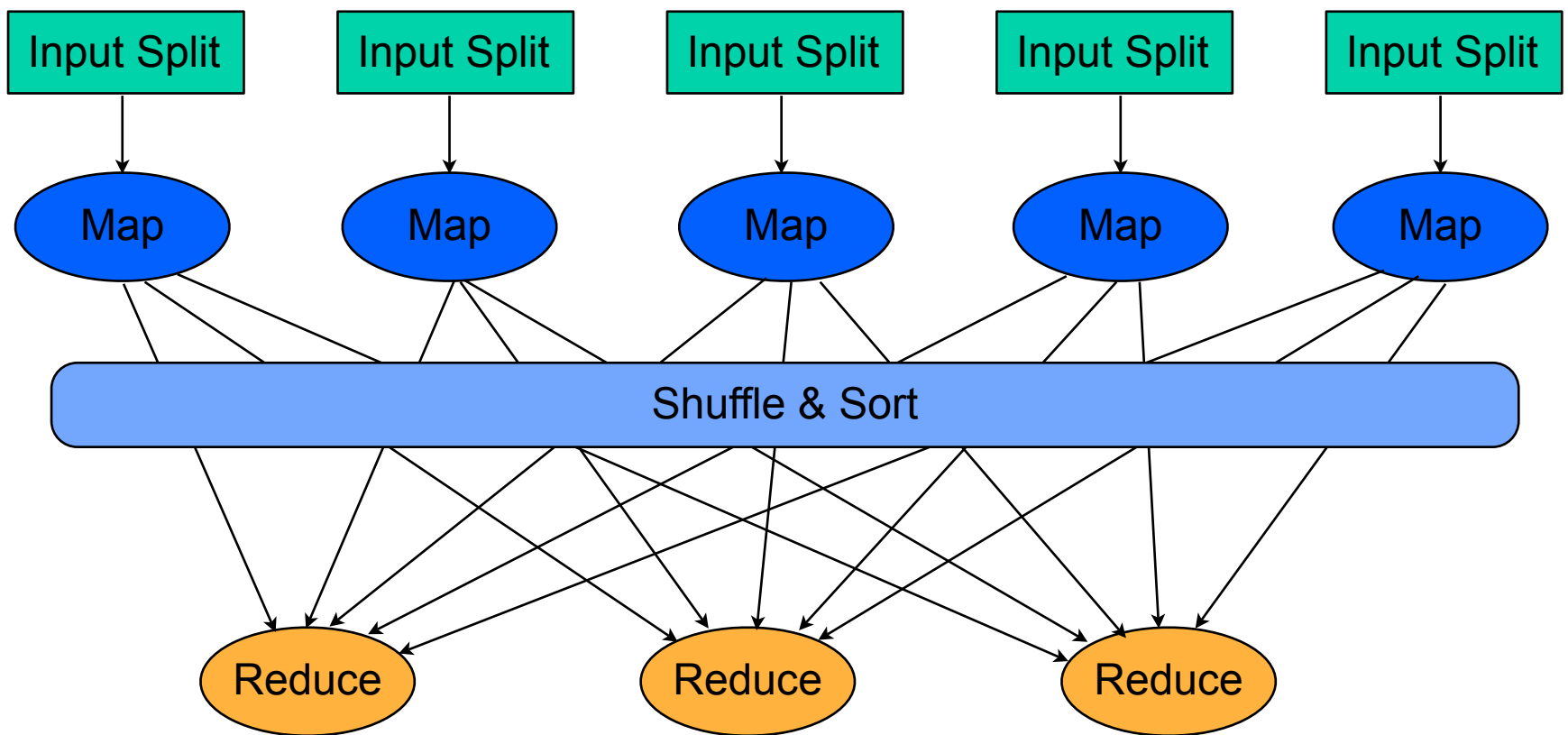
MapReduce



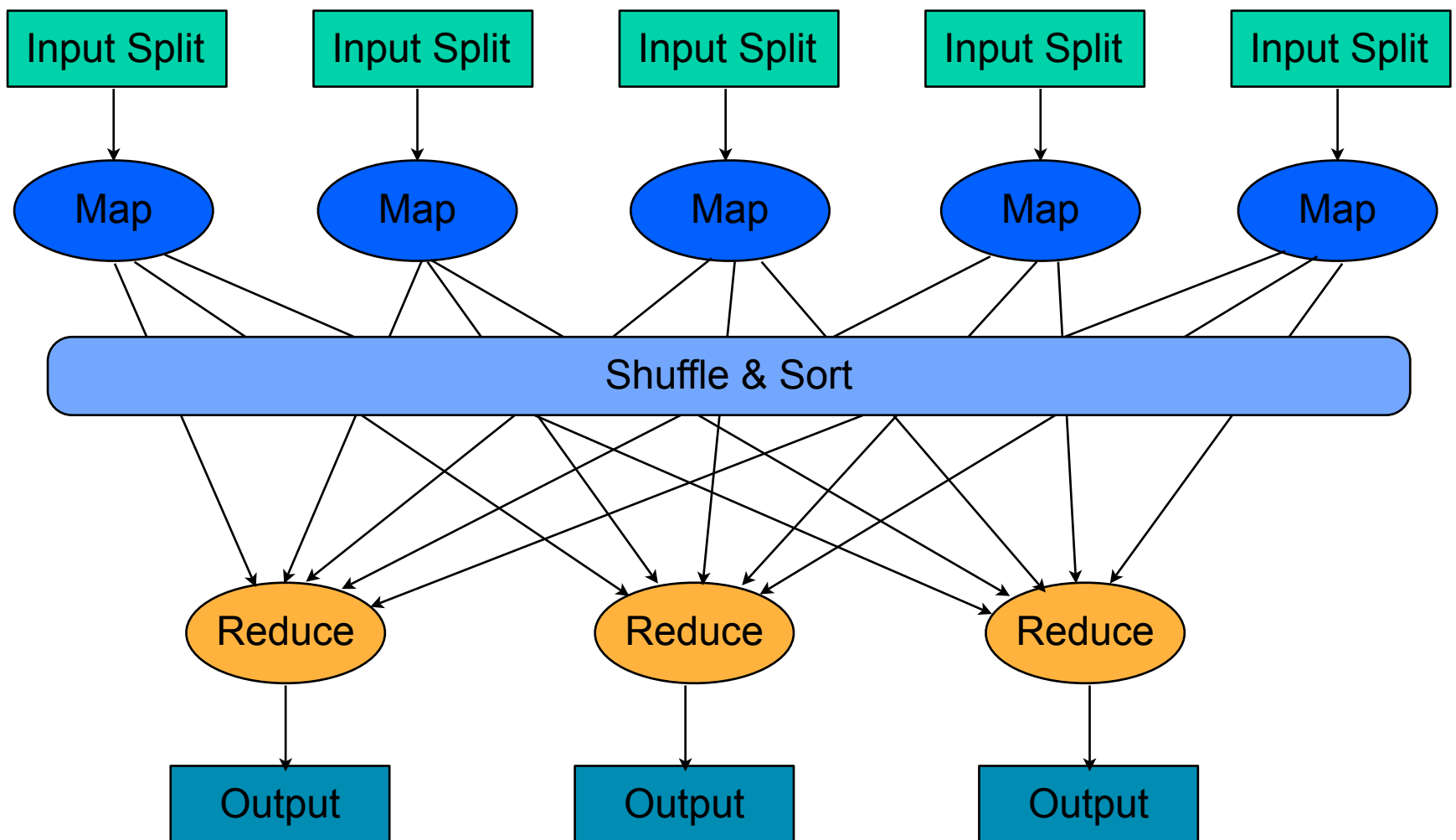
MapReduce



MapReduce



MapReduce



MapReduce (cont.)

- Application writer specifies
 - A set of input files
 - A pair of functions called *Map* and *Reduce*
 - Map transforms input records into (k_m, v_m) pairs
 - Reduce combines all (k_m, v_m) with same k_m into (k_r, v_r)
- Framework
 - All phases are distributed among many tasks
 - Allocates resources and schedules tasks on the cluster
 - Generates splits from input files, one per map task
 - Co-location of storage & computation
 - Shuffles & sort tuples according to their keys
 - Reliability: Handles node and task failures

MapReduce Example

- Input: multi-TB dataset
- Record: Vector with 3 float32_t values (v0, v1, v2)
- Goal: Plot count vs. value of v1
 - Frequency count for the values of v1
 - $V_{\min} < V_1 < V_{\max}$
 - 1000 buckets

MapReduce Example (cont.)

- Map function takes a single record $r=(v_0, v_1, v_2)$

```
Map(r) {  
  if (r.v1 < v_max && r.v1 > v_min) {  
    // Extract desired component v1  
    Emit((r.v1-v_min)/bucket_size, 1);  
  }  
}
```

- Reduce receives groups with the same k.

- Reduce (Key k, Iterator values) {
 sum = 0;
 while (iterator.next()) {
 sum += iterator.getValue();
 }
 emit(k*bucket_size+v_min, sum);
}

HDFS: Hadoop Distributed File System

- Open source counterpart of the Google file system
- Fault tolerant, scalable, distributed storage system
- Stores very large files across a large machine cluster
- Files are divided into uniform sized blocks and distributed across cluster nodes
- Blocks are replicated to handle hardware failure
- Corruption detection and recovery: FS-level checksumming
- HDFS exposes block placement:
 - Enables moving computation to data

Getting Started

- Goal: Get faculty & students active in DISC
- Hardware: Rent from Amazon
 - Elastic Compute Cloud (EC2)
 - Generic Linux cycles for \$0.10 / hour (\$877 / yr)
 - Simple Storage Service (S3)
 - Network-accessible storage for \$0.15 / GB / month (\$1800/TB/yr)
 - Example: maintain crawled copy of web
50 TB, 100 processors, 0.5 TB/day refresh ~ \$250K / year



- Software

- Hadoop Project
 - Open source project providing file system and MapReduce
 - Supported and used by Yahoo
 - Prototype on single machine, map onto cluster



Rely on Kindness of Others

Press Release 08-031

NSF Partners With Google and IBM to Enhance Academic Research Opportunities

Computer science researchers at universities and colleges will be able to utilize large-scale computing cluster

February 25, 2008

Today the National Science Foundation's Computer and Information Science and Engineering (CISE) Directorate announced the creation of a strategic relationship with Google Inc. and IBM. The Cluster Exploratory (CluE) relationship will enable the

- Google setting up dedicated cluster for university use
- Loaded with open-source software including Hadoop
- IBM providing additional software support
- NSF administering through the CLUE program

More Sources of Kindness: Yahoo! M45

Yahoo, Carnegie Mellon Switch On Supercomputer



Submitted by [David A. Utter](#) on Mon, 11/12/2007 - 11:08.

 [Comment](#) |  [Email](#) |  [Print](#)

The M45 supercomputer provided by Yahoo opened its ports to its partners at Carnegie Mellon University, where the initiative should help boost research that benefits the broader Internet community.



For those of you firing up the old faithful laptop for a morning of surfing, blogging, maybe a little development work, get a load of what some of the lucky geeks at [Carnegie Mellon University](#) got to play with this morning:

The M45, Yahoo's supercomputing cluster, has approximately 4,000 processors, three terabytes of memory, 1.5 petabytes of disks, and a peak performance of more than 27 trillion calculations per second (27 teraflops), placing it among the top 50 fastest

- Yahoo! is a major supporter of Hadoop
- Yahoo! plans to work with other universities


Beyond the U.S.

March 24 2008

Yahoo, Tata Subsidiary In Research Pact

Duncan Riley

[9 comments >>](#)

Yahoo **has announced**  an agreement with Computational Research Laboratories (CRL, a wholly owned subsidiary of Indian conglomerate Tata) to jointly undertake cloud computing research.



Under the deal, CRL will give access to one of world's top five supercomputers "that has substantially more processors than any supercomputer currently available for cloud computing research."

Testbed for system research in DISC systems

HP, Yahoo and Intel Create Compute Cloud

[Stacey Higginbotham](#), Tuesday, July 29, 2008 at 10:37 AM PT

[Comments \(8\)](#)

Related Stories

[HP Weds Cloud and High-performance Computing](#)

[Intel Friends Facebook to Make x86 Chips Sexy](#)

[Elastra Gets \\$12M — Is It Amazon's Enterprise Play?](#)

Powered by [Sphere](#)

Updated at the bottom: At long last, Hewlett-Packard is stepping up with an answer to cloud computing by inking a partnership with two other big technology vendors and three universities to create a cloud computing testbed. Through its R&D unit, HP Labs, the computing giant had teamed up with Intel, Yahoo, the Infocomm Development Authority of Singapore (IDA), the University of Illinois at Urbana-Champaign, the National Science Foundation (NSF) and the Karlsruhe Institute of Technology in Germany.

M45 Projects

- Targeted web crawling
- Automatic analysis for grading document reading difficulty.
- Language N-gram extraction
- Grammar induction
- Statistical machine translation
- Large-scale graph mining
- Understanding Wikipedia collaboration
- Large-scale scene matching: Retrieve and process images
- Parallel file systems for Hadoop

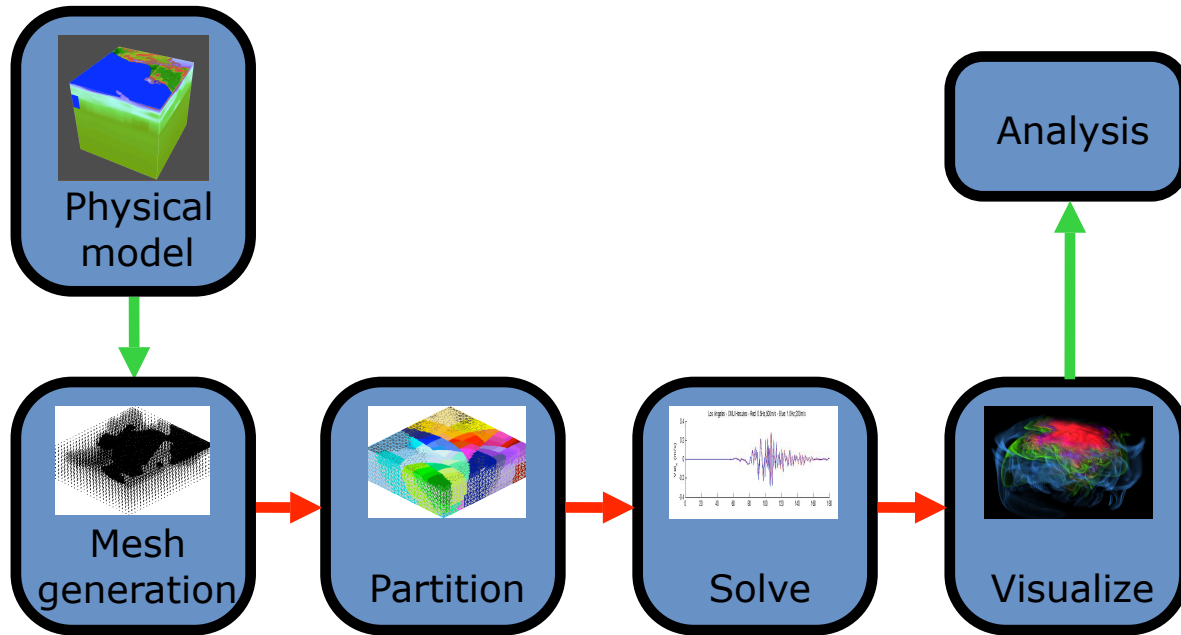
Science-related projects

- Earth Science related
 - Material ground model generation
 - Analysis of simulation-generated wavefields
 - Wavefield comparison
- Astrophysics
 - Large-scale Halo finding
 - Percolation analysis
 - N-point correlation functions
 - Image analysis and classification

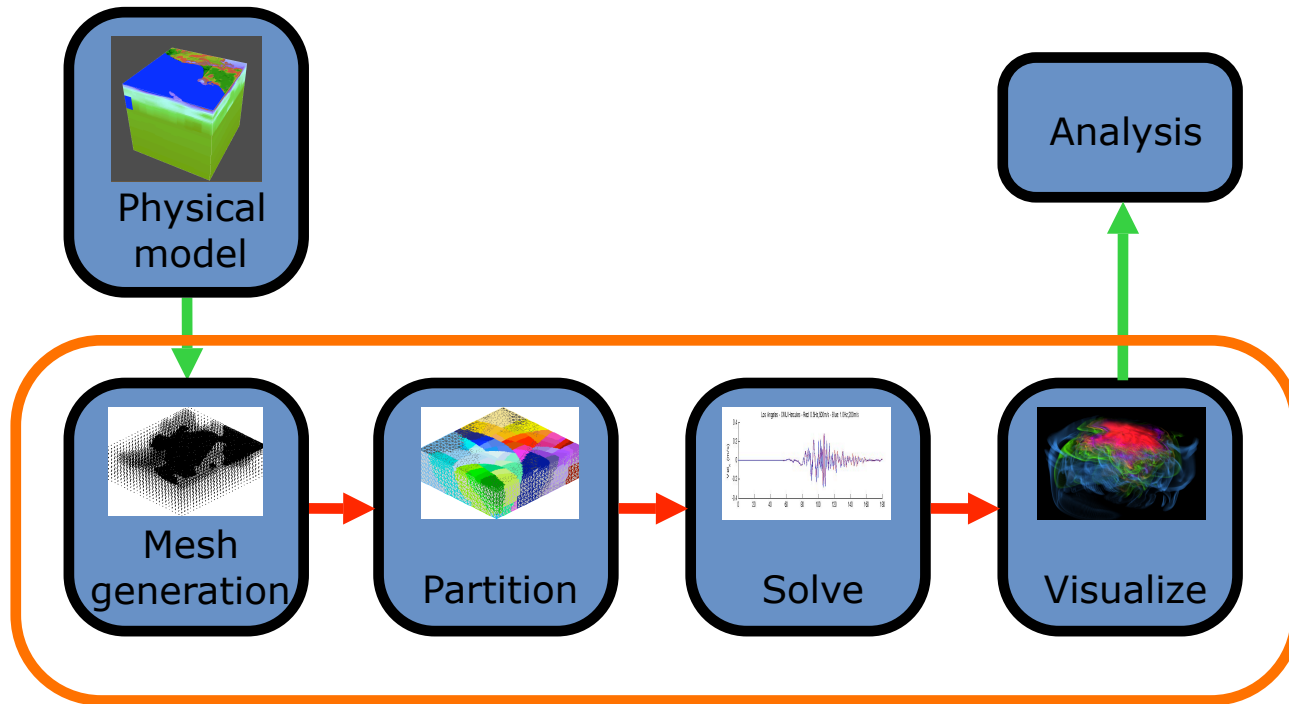
Science-related projects

- Earth Science related
 - Material ground model generation
 - Analysis of simulation-generated wavefields
 - Wavefield comparison
- Astrophysics
 - Large-scale Halo finding
 - Percolation analysis
 - N-point correlation functions
 - Image analysis and classification

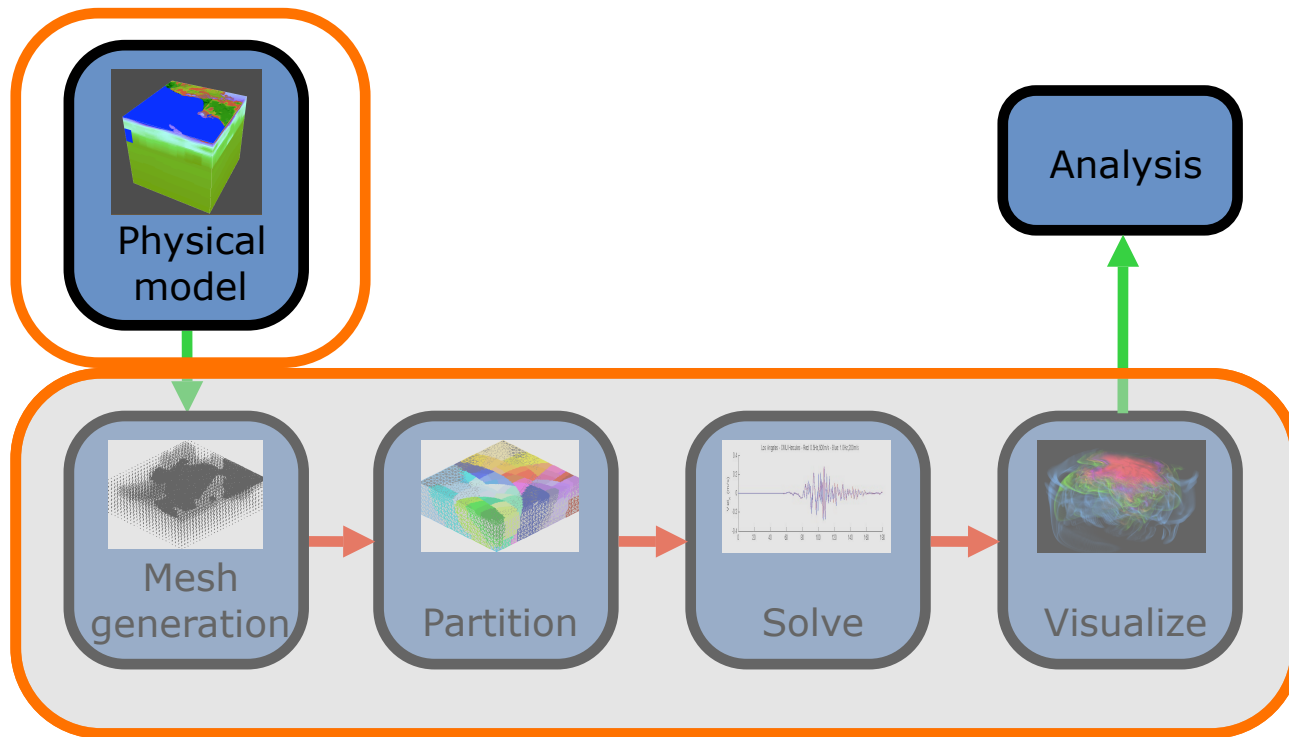
Ground motion modeling 101



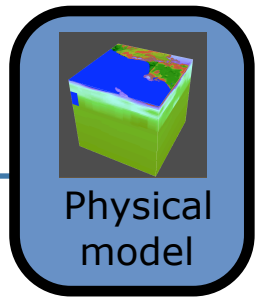
Ground motion modeling 101



Ground motion modeling 101



Ground model generation



- Populate an octree spatial structure with ground properties
- Octree has lower query cost (10 - 100X faster)
- Involves creating samples at high spatial resolution (10m)
- Samples are obtained from an external program
 - Reads: lat, lon, depth
 - Outputs: ground density and wave velocity (r, a, b)

Steve Schlosser, Michael Ryan, Julio López, Ricardo Taborda, Jacobo Bielak, David O'Hallaron.
"Generating ground models of Southern California", Supercomputing 2008

SCEC



Image credit: Amit Chourasia, Visualization Services, SDSC

SCEC

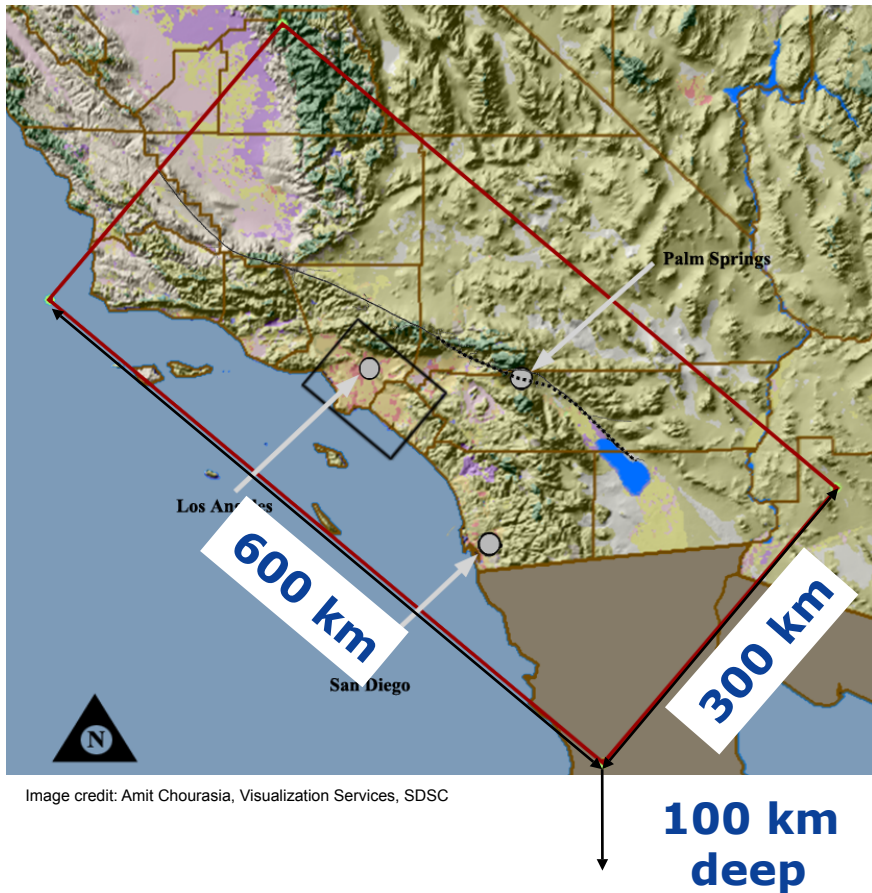


Image credit: Amit Chourasia, Visualization Services, SDSC

Goal:

Sample entire region at 10m resolution

$6 \times 10^4 \times 3 \times 10^4 \times 1 \times 10^4 = 18 \times 10^{12}$ sample points!

~1 PB of data uncompressed

Approach:

Reduce early and reduce often

Map/Reduce implementation

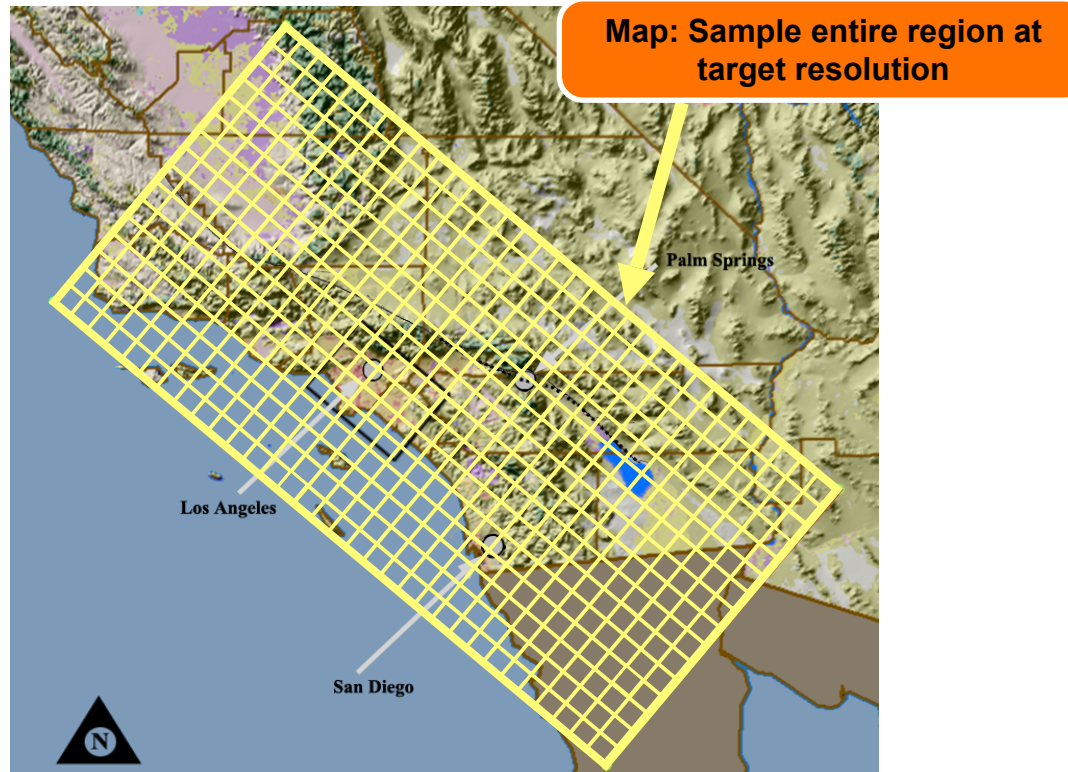
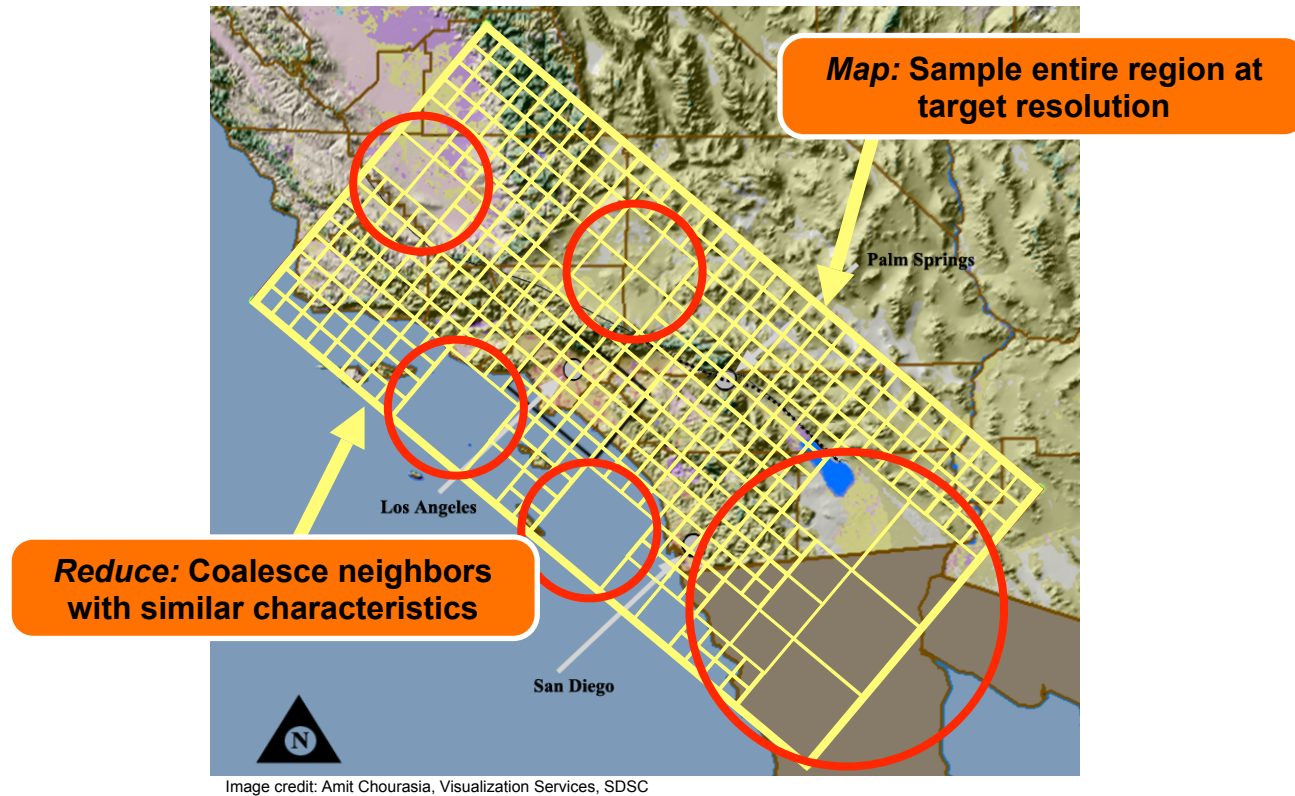
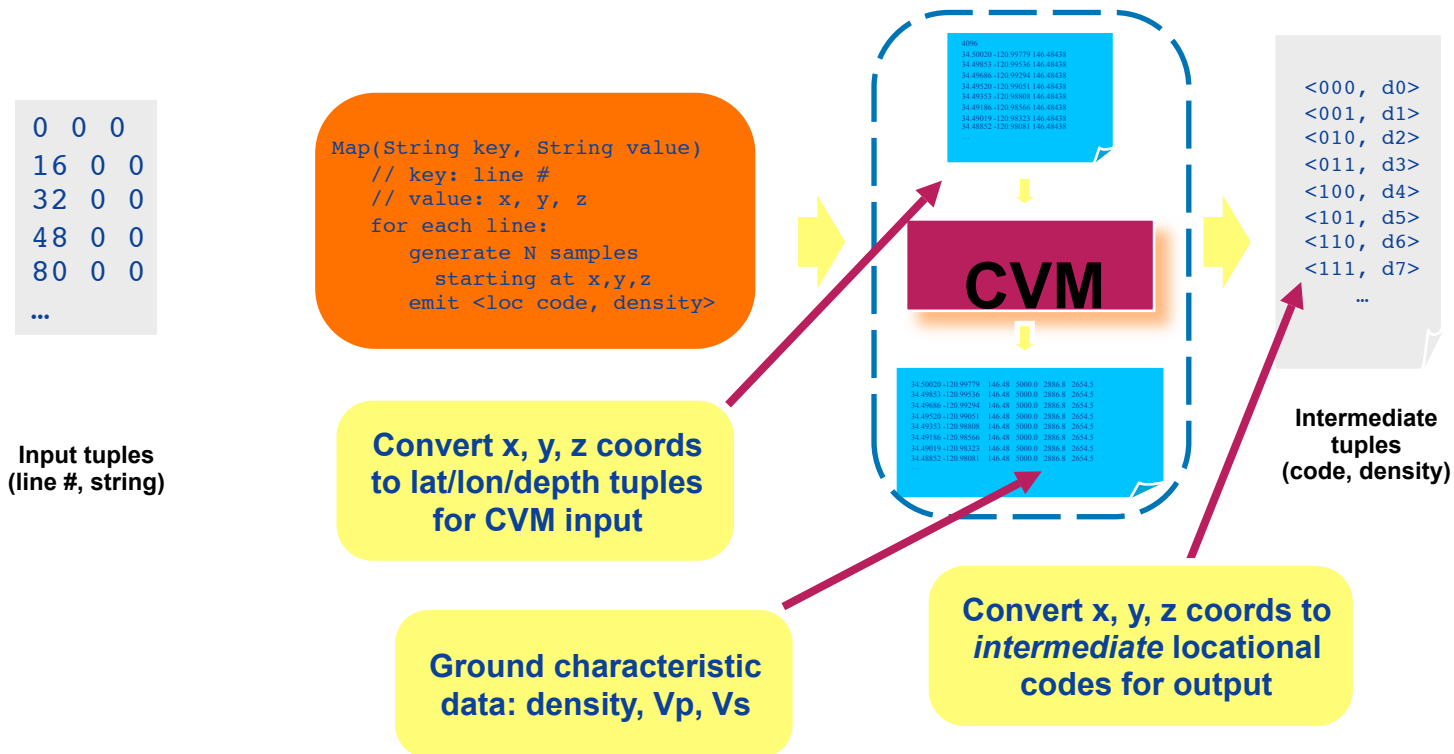


Image credit: Amit Chourasia, Visualization Services, SDSC

Map/Reduce implementation



Map implementation



Intermediate key manipulation

<000, d0>
<001, d1>
<010, d2>
<011, d3>
<100, d4>
<101, d5>
<110, d6>
<111, d7>
...

**Intermediate
tuples
(code, density)**



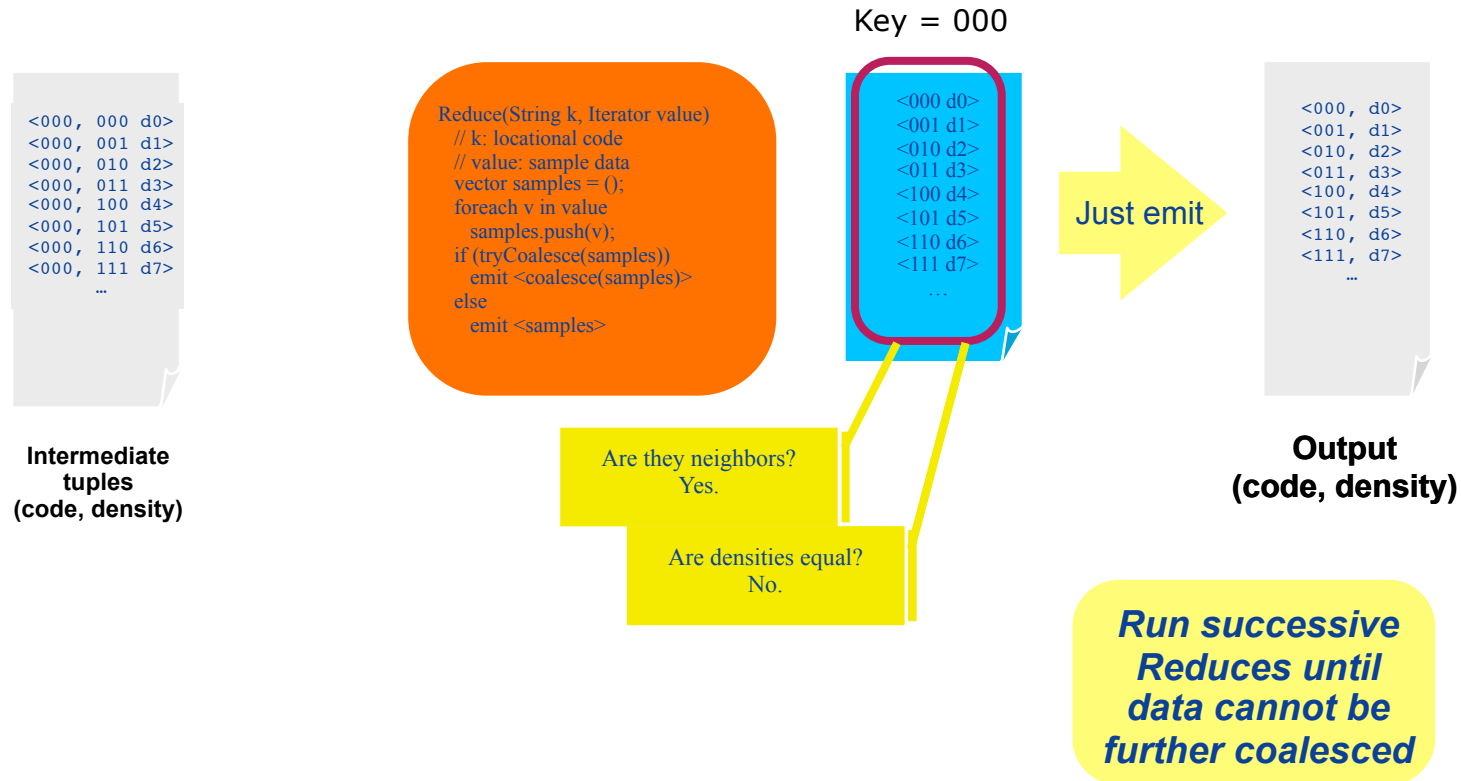
<000, 000 d0>
<000, 001 d1>
<000, 010 d2>
<000, 011 d3>
<000, 100 d4>
<000, 101 d5>
<000, 110 d6>
<000, 111 d7>
...

**Manipulated
tuples
(code, density)**

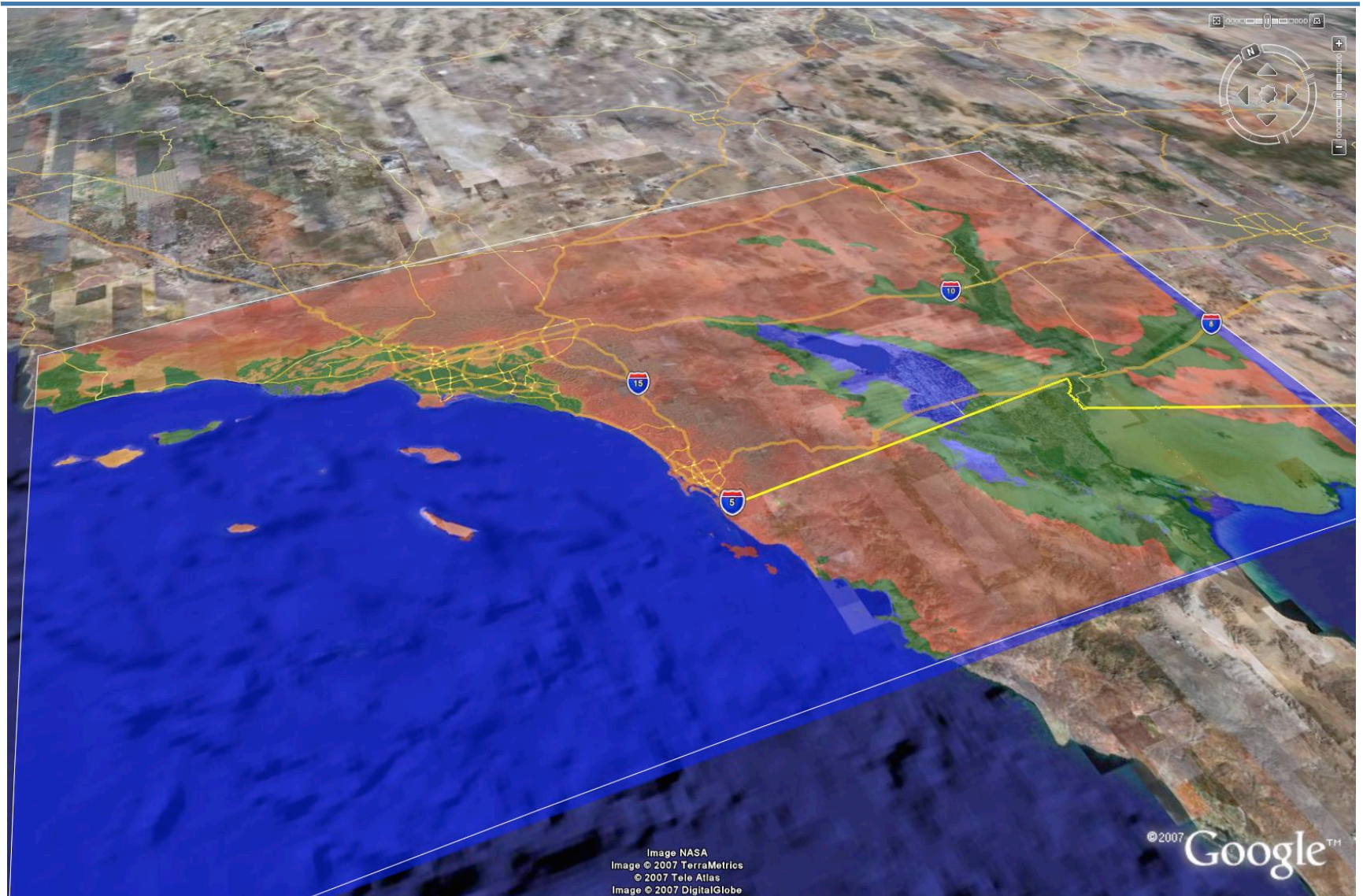
Clear 3 low-order bits per
octree level

Naturally gathers neighboring
tuples together for Reduce

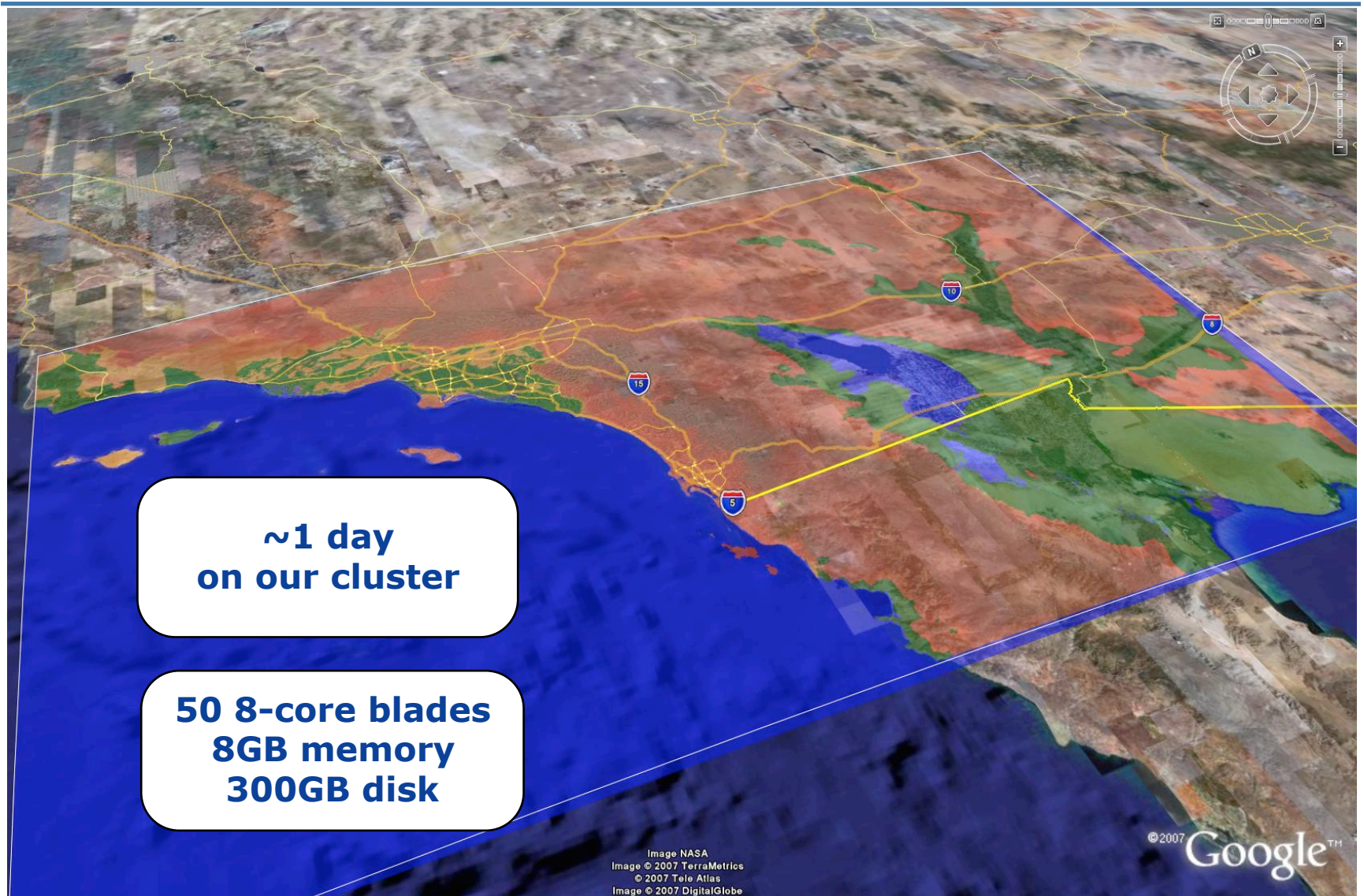
Reduce implementation



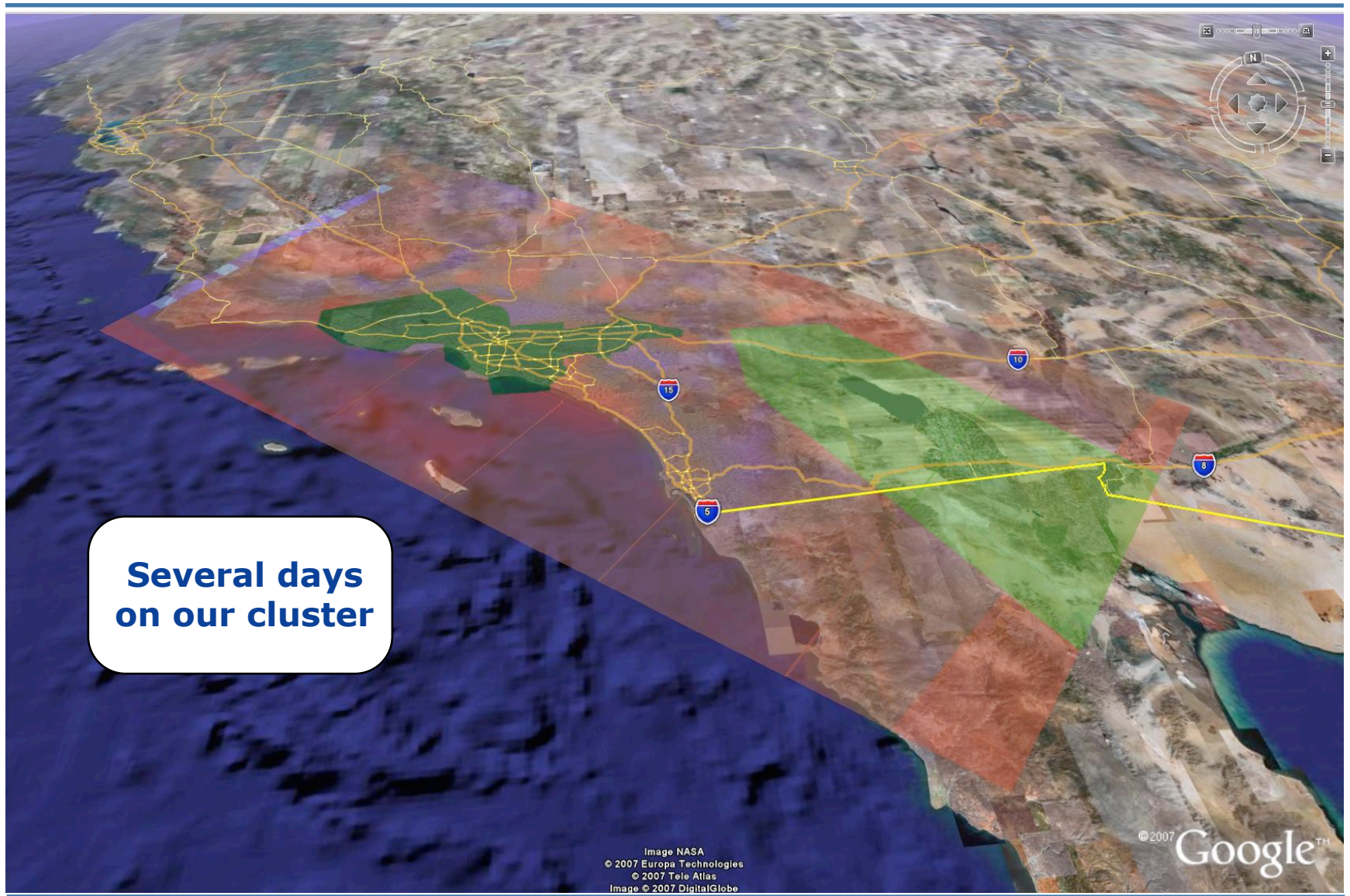
Harvard



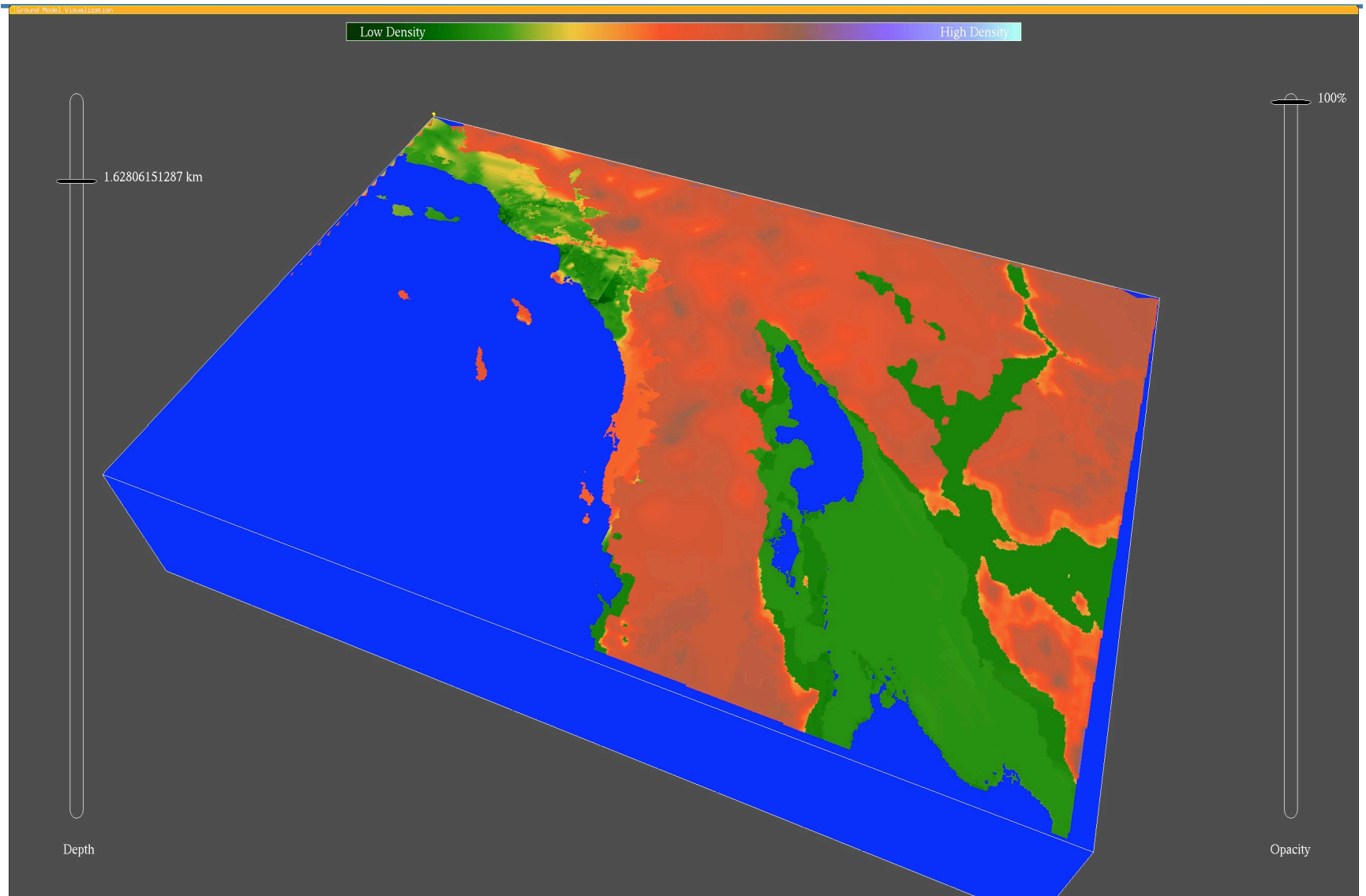
Harvard



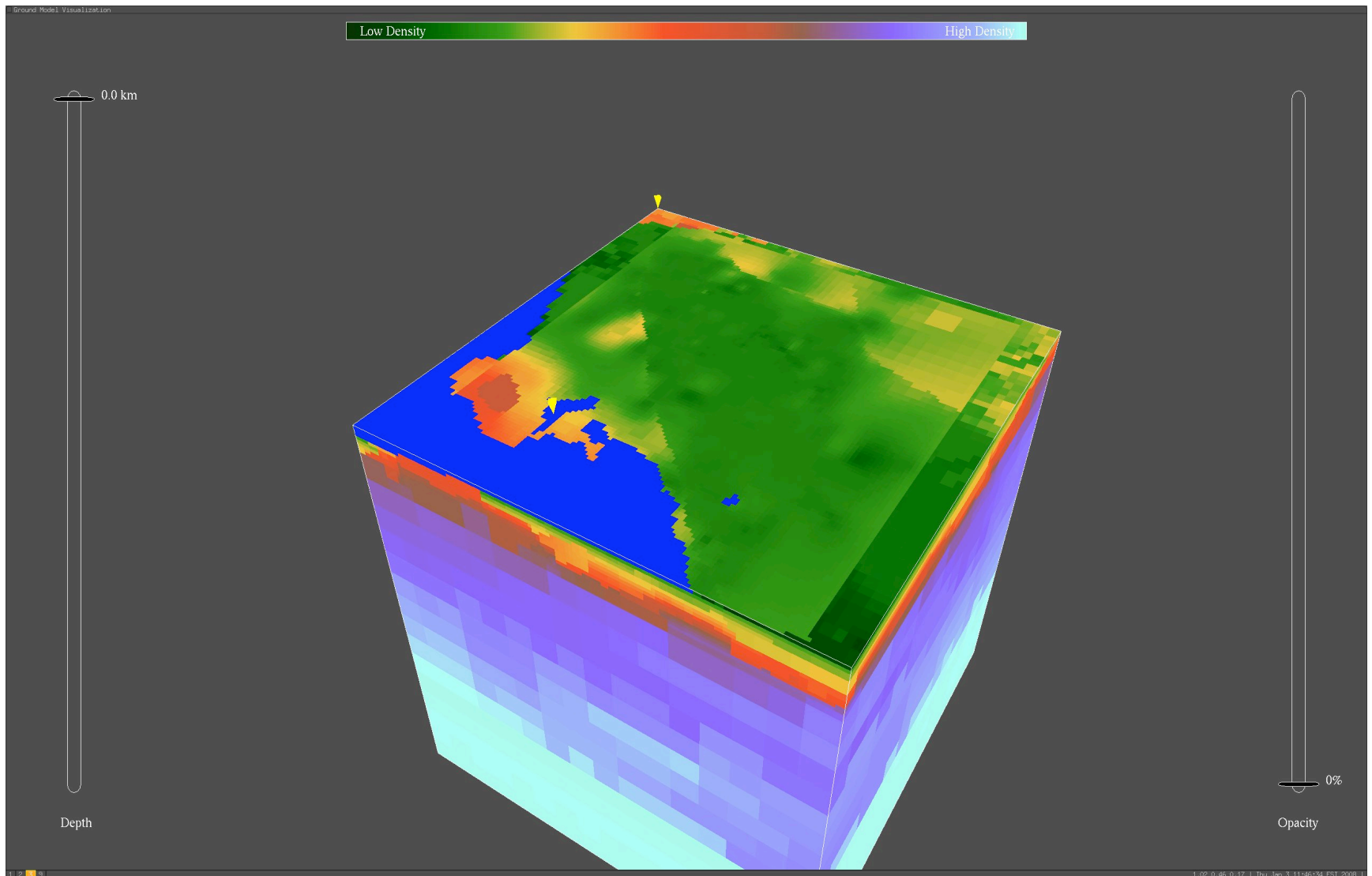
SCEC



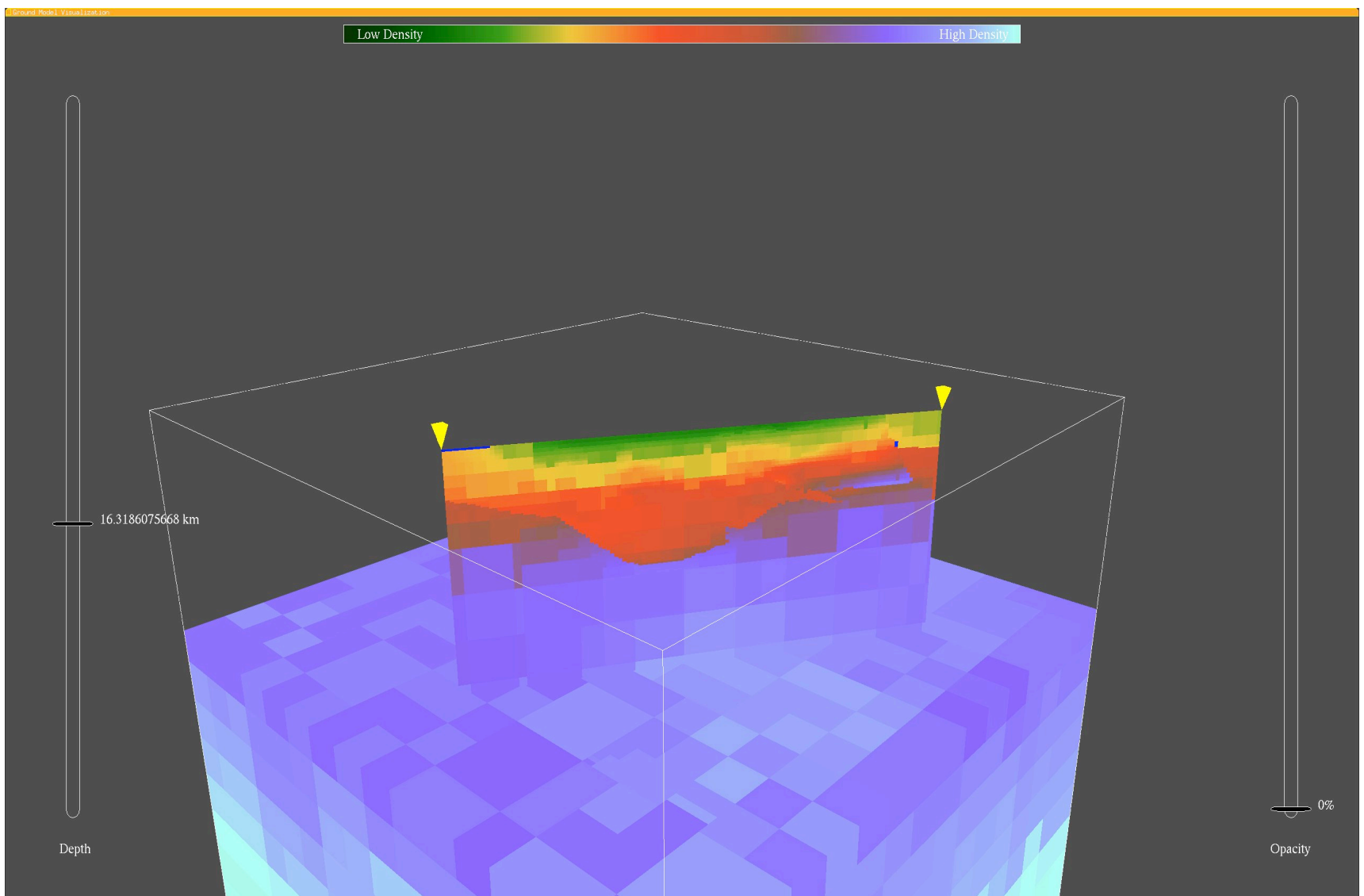
Harvard



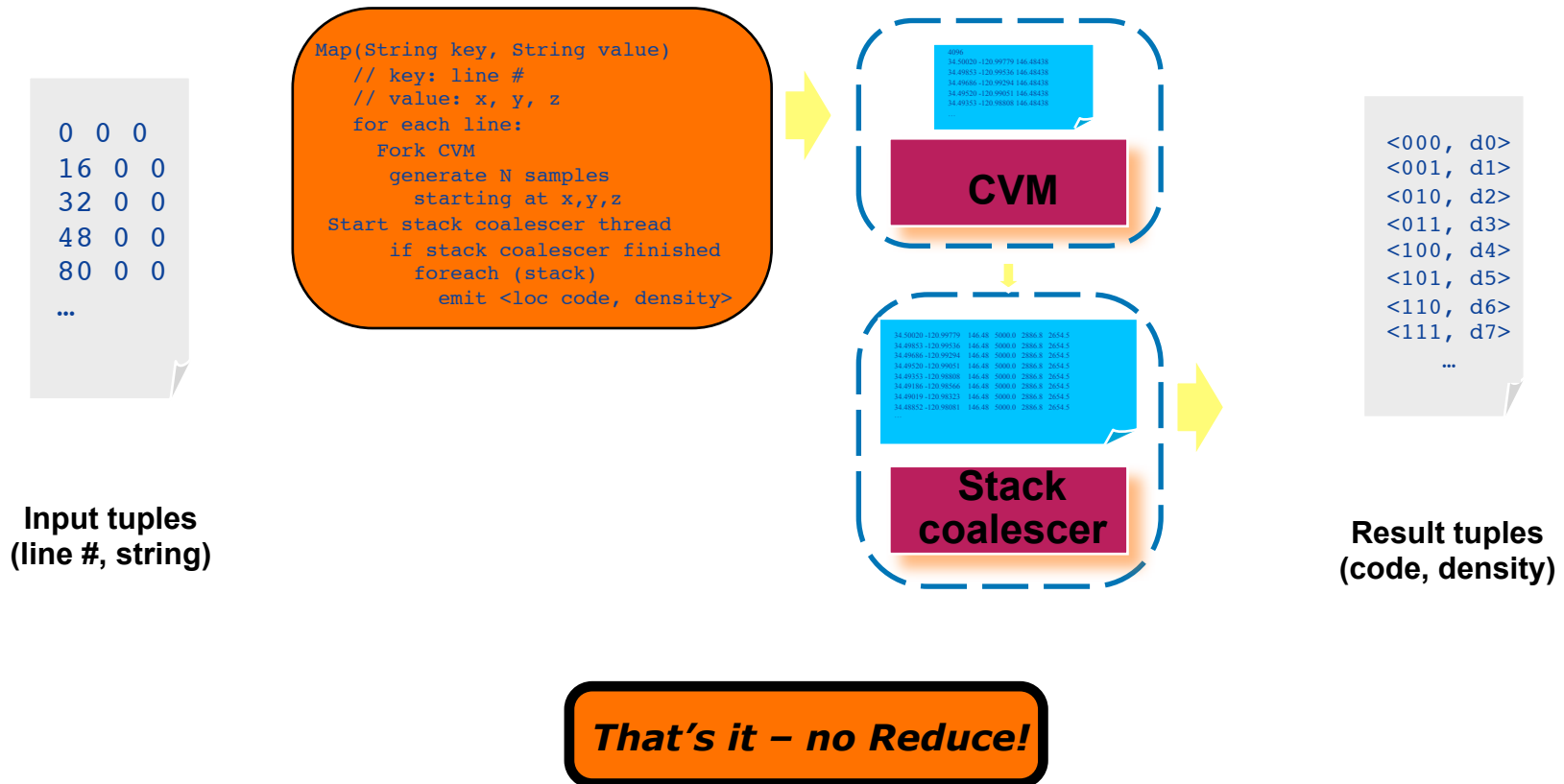
Harvard



Harvard



Stack-based coalescing



Ground Model Generation Summary

- Used Hadoop to build a ground model generator
- Hadoop implementation runs in $O(\text{days})$
- Stack-based Hadoop and C versions run in several hours
- Cost of distributed group-by are not necessary for this app
- Hadoop hides a lot of complexity

Hadoop: What we've learned

- There's a learning curve:
 - Programming: how to plug things together
 - How to mix existing legacy code & new
 - How to configure Hadoop
 - Being good web crawlers, experiential learning
- Dealing with the input: formats, small files, etc.
- **Achieved good problem size scaling**
... in a short period of time.

MapReduce & Hadoop strengths

- Simple, easy-to-understand programming model
- Good for unstructured data: customized parsing
- Powerful “GROUP BY” primitive
 - Unordered input
 - Suited for computing statistics, e.g., term frequency
- System - application separation
 - Distributed and out-of-core processing
 - Resilient failure handling
 - Enables co-location of storage and computation

Shortcomings

- Low-level primitive for some application domains
 - Need higher-level abstractions
- Constraining pattern: M/S/R, M-only
 - What about recursive block transformations?
- Coarse-grained lockstep operations
 - No coordination between tasks, no explicit RPC
- Little benefit for ordered data
- Cumbersome multi-dataset operations
- Reading custom data formats

Desiderata for DISC Systems

- **Focus on Data**
 - Terabytes, not tera-FLOPS
- **Problem-Centric Programming**
 - Platform-independent expression of data parallelism
- **Interactive Access**
 - From simple queries to massive computations
- **Robust Fault Tolerance**
 - Component failures are handled as routine events

CS Research Issues

- Applications
 - Astroinformatics, language translation, image processing...
- Application Support
 - Machine learning over very large data sets
 - Web crawling
- Programming
 - Programming models to support large-scale computation
 - Distributed databases
- System Design
 - Error detection & recovery mechanisms
 - Resource scheduling and load balancing
 - Distribution and sharing of data across system

Choosing Execution Models

- **Message Passing / Shared Memory**
 - Achieves high performance when everything works well
 - Requires careful tuning of programs
 - Vulnerable to single points of failure
- **Map/Reduce**
 - Allows for abstract programming model
 - More flexible, adaptable, and robust
 - Performance limited by disk I/O
- **Alternatives?**
 - Is there some way to combine to get strengths of both?
 - Other models such as MSR Dryad.

Concluding Thoughts

- Need for a new approach to large-scale computing
 - Optimized for data-driven applications
 - Technology favoring centralized facilities
 - Storage capacity & computer power growing faster than network and I/O bandwidth
- Industry is catching on quickly
 - Large crowd for Hadoop Summit
 - Quick adoption by many companies
- University researchers / educators getting involved
 - Spans wide range of CS disciplines
 - Across multiple institutions

More Information

Data-Intensive Scalable Computing

<http://www.pdl.cmu.edu/DISC>